

Form Control Arguments:- First argument is reserved for the values (52)

app.component.ts

```
export class AppComponent
```

```
{ title = 'reactiveOnline class';
```

```
  form = new FormGroup({
```

```
    name : new FormControl("lavanya"),
```

```
    email : new FormControl("lavanya@gmail.com"),
```

```
    password : new FormControl("pass")
```

```
  })
```

```
  signUp()
```

```
  {
```

```
    console.log(this.form)
```

```
  }
```

```
}
```

Output:-

Name

lavanya

Email

lavanya@gmail.com

Password

pass

Submit

app.component.ts

```
export class AppComponent {
```

```
  title = 'reactive Online';
```

```
  form = new FormGroup({
```

```
    username: new FormControl("", [Validators.required,  
    Validators.minLength(3), Validators.maxLength(15)])  
  })
```

```
  get un()
```

```
  {
```

```
    return this.form.get('username')
```

```
  }
```

```
  fun()
```

```
  {
```

```
    console.log(this.form)
```

```
  }
```

app.component.html:

```
<h1> reactive form </h1>
```

```
<div class="container">
```

```
  <form [formgroup]="form" (ngSubmit)="fun()">
```

```
    <div class="form-group">
```

```
      <label> Password </label>
```

```
      <input type="text" class="form-control"
```

```
        formControlName="username">
```

```
    <div *ngIf="un.touched && un.invalid">
```

```
      <div *ngIf="un.errors.required"> password required </div>
```

```
      <div *ngIf="un.errors.minlength"> min length is 3 </div>
```

```
      <div *ngIf="un.errors.maxlength"> max length is 15 </div>
```

```
    </div>
```

```
  </div>
```

↓
class = "text-danger"

```
  <button class="btn btn-primary" type="submit"> Submit  
    </button>
```

```
</form>
```

```
</div>
```

Output:-

Case (1)

Password

password is required

Submit

Case (3):-

Password

max length is 15.

Submit

Case (2):-

Password

min length is 3

Submit

- * `get()` returns the particular property (with respect to username we have created on as the reference).
- * Template driven form and reactive form both has same methods and properties we are invoking with the help of reference (object)